

Unclassified

## UGV Interoperability Profile (IOP) – Overarching Profile Custom Services, Messages, and Transports Version 0



Robotic Systems, Joint Project Office (RS JPO)  
SFAE-GCS-UGV MS 266  
6501 East 11 Mile Road  
Warren, MI 48397

21 December 2011

UNCLASSIFIED: Distribution Statement A. Approved for public release.

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE <b>21 DEC 2011</b>		2. REPORT TYPE		3. DATES COVERED <b>00-00-2011 to 00-00-2011</b>	
4. TITLE AND SUBTITLE <b>UGV Interoperability Profile (IOP) - Overarching Profile Custom Services, Messages, And Transports, Version 0</b>				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <b>Robotic Systems, Joint Project Office (RS JPO),SFAE-GCS-UGV MS 266,6501 East 11 Mile Road,Warren,MI,48397</b>				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT <b>Approved for public release; distribution unlimited</b>					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT <b>Same as Report (SAR)</b>	18. NUMBER OF PAGES <b>52</b>	19a. NAME OF RESPONSIBLE PERSON
a. REPORT <b>unclassified</b>	b. ABSTRACT <b>unclassified</b>	c. THIS PAGE <b>unclassified</b>			

## Table of Contents

1	Scope.....	1
2	Source Documents.....	2
2.1	Government Documents .....	2
2.2	Non-Government Documents.....	2
3	Custom Services .....	3
3.1	Leader Management Service .....	3
3.1.1	Description.....	3
3.1.2	Vocabulary.....	4
3.1.3	Encoding.....	4
3.1.4	Protocol Behavior .....	7
3.2	Leader Follower Driver .....	8
3.2.1	Description.....	8
3.2.2	Assumptions .....	9
3.2.3	Vocabulary.....	9
3.2.4	Encoding.....	9
3.2.5	Protocol Behavior .....	14
3.3	Communicator Service .....	15
3.3.1	Description.....	16
3.3.2	Assumptions .....	16
3.3.3	Vocabulary.....	16
3.3.4	Encoding.....	16
3.3.5	Protocol Behavior .....	24
3.4	Platform Mode Service .....	26
3.4.1	Description.....	27
3.4.2	Assumptions .....	27
3.4.3	Vocabulary.....	27
3.4.4	Encoding.....	27
3.4.5	Protocol Behavior .....	30
3.5	Health Monitor Service .....	30
3.5.1	Description.....	31
3.5.2	Assumptions .....	31
3.5.3	Vocabulary.....	31
3.5.4	Encoding.....	31

## Unclassified

3.5.5	Protocol Behavior .....	36
3.6	Health Reporter Service .....	36
3.6.1	Description.....	37
3.6.2	Assumptions .....	37
3.6.3	Vocabulary.....	37
3.6.4	Encoding.....	38
3.6.5	Protocol Behavior .....	39
3.7	Digital Stream Discovery .....	40
3.7.1	Description.....	41
3.7.2	Assumptions .....	41
3.7.3	Vocabulary.....	41
3.7.4	Encoding.....	41
3.7.5	Protocol Behavior .....	43
3.8	Preset Pose Service.....	44
3.8.1	Description.....	45
3.8.2	Assumptions .....	45
3.8.3	Vocabulary.....	45
3.8.4	Encoding.....	45
3.8.5	Protocol Behavior .....	46
4	Custom Messages .....	47
5	Custom Transports.....	47
6	Conformance and Validation Requirements.....	48
7	Appendix A – Acronyms and Abbreviations .....	49

## 1 Scope

This document is an attachment to the *UGV Interoperability Profile (IOP) Overarching Profile* and serves the purpose of compiling all custom services, messages, and transports into a single location. These services, messages and transports are configuration controlled via the UGV IOP IPT and approved for use by the UGV IOP Executive Board.

Custom services and messages will be published and distributed to the RS JPO stakeholder community without proprietary markings. Custom messages that cannot meet this distribution will not be specified for use within the UGV IOP.

## **2 Source Documents**

The following documents/data items were utilized as reference source material in the conduction of this domain analysis.

### **2.1 *Government Documents***

- N/A

### **2.2 *Non-Government Documents***

- N/A

### 3 Custom Services

Custom services includes all services that have been created specifically to meet needs of the Interoperability Profile effort that are not already found in a published or draft SAE AS-4 JAUS document (or other standard). Fully created custom services declare and define the messages they use in this section. For custom messages attached to existing services, see Section 4 Custom Messages.

#### 3.1 *Leader Management Service*

name= LeaderManagementService

version=0.1

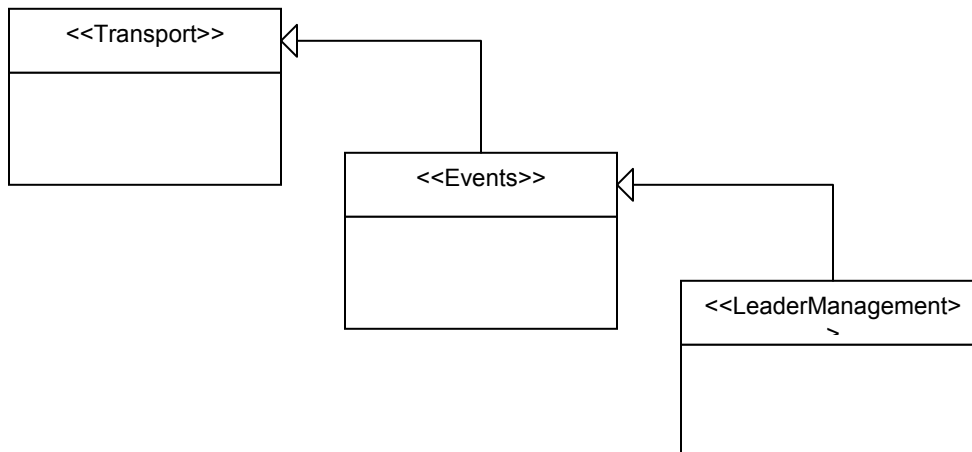
id=urn:jpo:autonomy: LeaderManagementService

Inherits-from Events

name=events

id= urn:jaus:jss:core: Events

version=1.0



**Figure 3.1-1: Leader Management Service Inheritance Diagram**

##### 3.1.1 Description

The Leader Management Service is intended to be hosted by the lead subsystem in a leader/follower operation. The Service allows followers to register with the leader, and requires periodic requests from the followers to maintain that registration. Furthermore, followers may request that the lead vehicle slow down or speed up to improve the leader/follower performance by using the speed override message.

Assumptions:

Messages may be delayed, lost or reordered.

### 3.1.2 Vocabulary

**Table 3.1-1: LEADER MANAGEMENT SERVICE VOCABULARY**

Message ID (hex)	Name	Command
<b>Input Set</b>		
FFD3	<a href="#">QueryFollowers</a>	false
FFD2	<a href="#">RegisterFollower</a>	false
FFD1	<a href="#">RequestSpeedOverride</a>	false
<b>Output Set</b>		
FFD4	<a href="#">ReportFollowers</a>	False
FFD5	RegisterFollowerResponse	False

**Table 3.1-2: LEADER MANAGEMENT SERVICE INTERNAL EVENTS SET**

Name	Interpretation
<i>RegistrationTimeout</i>	Occurs when registration is not re-acquired within the required timeout period

### 3.1.3 Encoding

#### 3.1.3.1 Input Set

##### 3.1.3.1.1 ID FFD3: QueryFollowers

Queries for a list of all active followers.

**Table 3.1-3: QUERY FOLLOWERS MESSAGE ENCODING**

Empty message body
--------------------

##### 3.1.3.1.2 ID FFD2: RegisterFollower

This message allows a follower to register with the lead subsystem, or disconnect (cancel) a previous registration. While such registration is not required for one subsystem to follow another, only subsystems that have registered with the leader can effect speed changes through the override mechanism.

**Table 3.1-4: REGISTER FOLLOWER MESSAGE ENCODING**

<div>body</div> <div>└ record name = FollowerRec</div>
<b>Record Name = FollowerRec</b>



## Unclassified

Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> RequestType	unsigned byte		false	Enum 0: CONNECT Enum 1: DISCONNECT

### 3.1.3.1.3 ID FFD1: RequestSpeedOverride

This message allows a follower to request an override to the lead subsystem's speed. The override may be given as a percentage, such that Actual Speed = Override \* Original Speed, or as an absolute value.

**Table 3.1-5: REQUEST SPEED OVERRIDE MESSAGE ENCODING**

<div style="text-align: right; padding-right: 20px;"> body  └─ <b>variant</b> name=OverrideType            <b>record</b> name=<a href="#">PercentRec</a>            <b>record</b> name=AbsoluteRec </div>					
<b>Record Name = PercentRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> SpeedOverride	unsigned byte	one	false	Percent. Scale Range [0...100].
<b>Record Name = AbsoluteRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> SpeedOverride	unsigned short integer	meters per second	false	Scaled Integer Lower Limit= 0 Upper Limit= 327.67

### 3.1.3.2 Output Set

#### 3.1.3.2.1 ID FFD4: ReportFollowers

This message reports a list of all followers registered with a leader.

**Table 3.1-6: REPORT FOLLOWERS MESSAGE ENCODING**

<div> <div>body</div> <div> <div>list name=FollowerList</div> <div>(count_field = unsigned byte)</div> <div>record name=<a href="#">FollowerRec</a></div> </div> </div>					
<b>Record Name = FollowerRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<bit_field> Follower_ID	unsigned integer	bit_field	false	Bits 0..7 : Component ID of Leader Bits 8..15 : Node ID of Leader Bits 16..31 : Subsystem ID of Leader

#### 3.1.3.2.2 ID FFD5: RegisterFollowerResponse

This message is sent as a response to a register request, or may be sent asynchronously if the follower does not periodically resend a register request. The required periodic rate is specified in this message.

**Table 3.1-7: REGISTER FOLLOWER RESPONSE MESSAGE ENCODING**

<div> <div>body</div> <div> <div>record name = FollowerRec</div> </div> </div>					
<b>Record Name = FollowerRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Result	unsigned byte		false	Enum 0: CONNECTED Enum 1: DISCONNECTED
2	<fixed_field> Timeout	unsigned byte	seconds	false	The follower must resend a Register request before the

					timeout elapses; otherwise, the leader may consider the follower as having disconnected.
--	--	--	--	--	--

### 3.1.4 Protocol Behavior

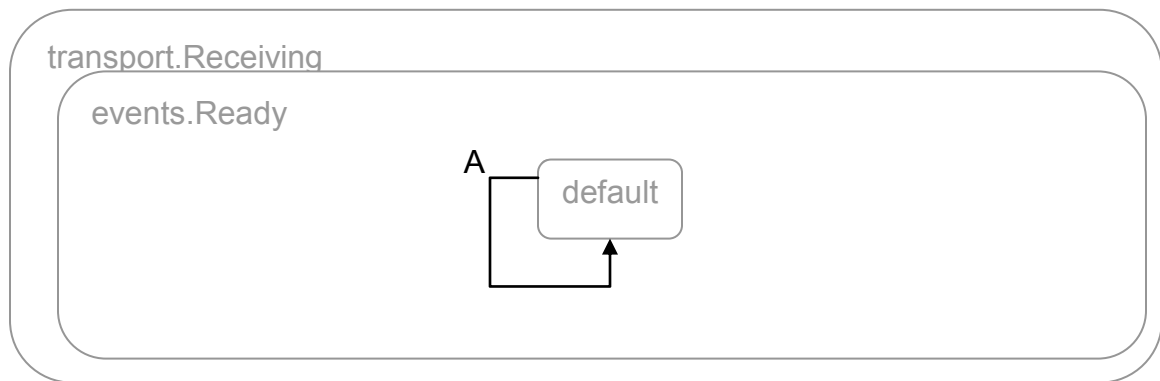


Figure 3.1-2: LEADER MANAGEMENT SERVICE PROTOCOL BEHAVIOR

Table 3.1-8: LEADER MANAGEMENT SERVICE STATE TRANSITION TABLE

Label	Trigger	Conditions	Actions
A	QueryFollowers		SendResponse ( 'ReportFollowers' )
	RegisterFollower		AddFollowerToRegistrationList SendResponse( 'RegisterFollowerResponse', CONNECTED )
	RequestSpeedOverride	<i>isRegistered</i>	<i>setOverride</i> ( msg )
	<i>RegistrationTimeout</i>		SendResponse( 'RegisterFollowerResponse', DISCONNECTED ) RemoveFollowerToRegistrationList

Table 3.1-9: LEADER MANAGEMENT SERVICE CONDITIONS

Condition	Interpretation
<i>isRegistered</i>	True if the component that sent the message is in the registered follower list

Table 3.1-10: LEADER MANAGEMENT SERVICE TRANSITION ACTIONS

Action	Interpretation
SendResponse	Send the specified response message

## Unclassified

AddFollowerToRegistrationList	Add the component that sent the request to the follower list
RemoveFollowerToRegistrationList	Remove the component from the follower list
setOverride	Update the leader's override value based on the new request. The current override is the minimum requested value of all registered followers.

### 3.2 Leader Follower Driver

name= LeaderFollowerDriver

version=0.1

id=urn:jpo:autonomy:LeaderFollowerDriver

Inherits-from Management

name=management

id= urn:jaus:jss:core: Management

version=1.0

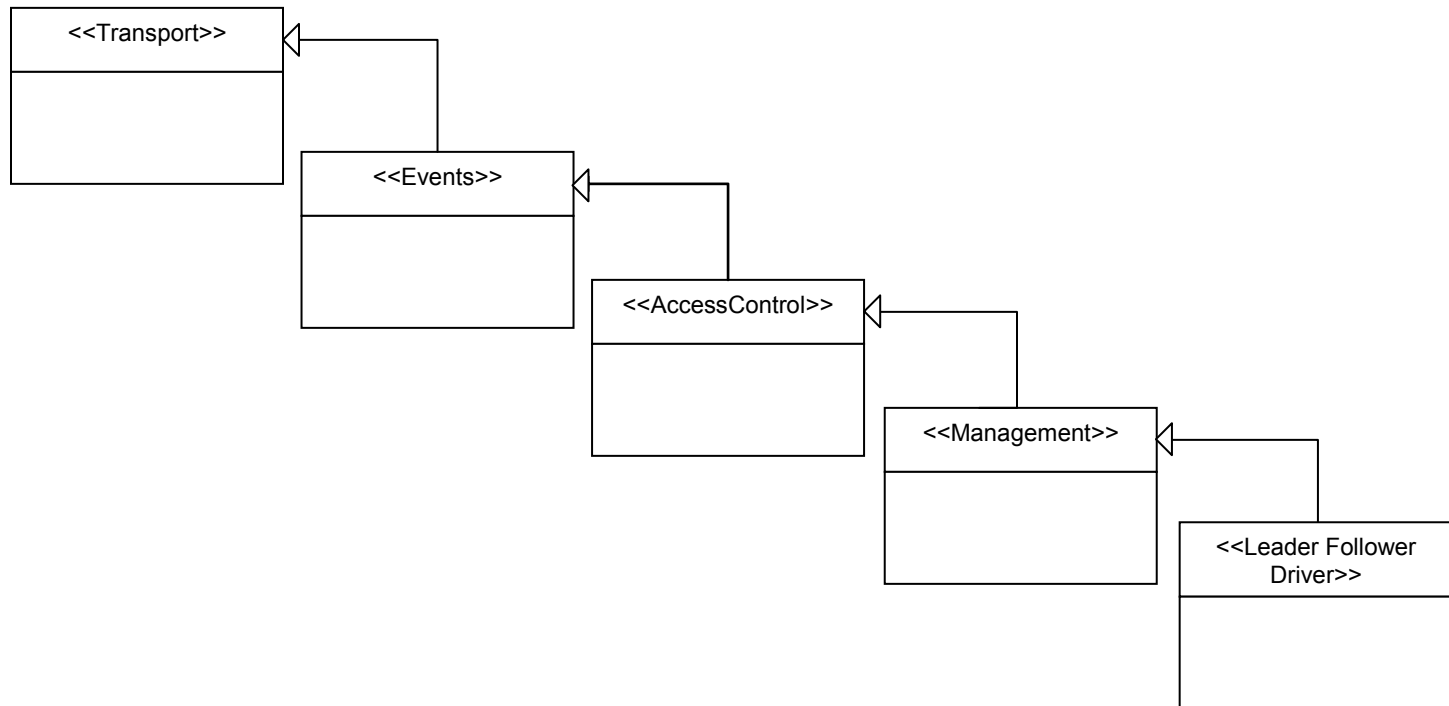


Figure 3.2-1: LEADER FOLLOWER DRIVER SERVICE

#### 3.2.1 Description

The Leader Follower Driver Service provides a mechanism for following the path of a leader. A leader can be identified by the ID of its Global Pose Sensor Service, or may be implicitly known by the implementation. In addition, a leader may host the Leader Management Service, which

allows a follower vehicle to adjust the speed of the lead vehicle to compensate for delays encountered, such as local obstacles, traffic conditions, etc.

### 3.2.2 Assumptions

Messages may be delayed, lost or reordered.

### 3.2.3 Vocabulary

Table 3.2-1: LEADER FOLLOWER DRIVER SERVICE VOCABULARY

Message ID (hex)	Name	Command
<b>Input Set</b>		
FFF3	<a href="#">QueryFollowerConfiguration</a>	false
FFF2	<a href="#">SetFollowerConfiguration</a>	true
FFF1	<a href="#">SetFollowerState</a>	true
<b>Output Set</b>		
FFF4	<a href="#">ReportFollowerConfiguration</a>	false

### 3.2.4 Encoding

#### 3.2.4.1 Input Set

##### 3.2.4.1.1 ID FFF3: *QueryFollowerConfiguration*

Queries the current state and configuration of a follower.

Table 3.2-2: QUERY FOLLOWER CONFIGURATION MESSAGE ENCODING

<pre> body └─ (empty) </pre>
Empty message body

##### 3.2.4.1.2 ID FFF2: *SetFollowerConfiguration*

This message sets the configuration for the follower. The leader is specified by an optional JAUS identifier that refers to a Global Pose Sensor Service hosted by the leader; if this JAUS identifier is not specified, the leader is assumed to be known a priori by the service. The optional offset values specify the follow behavior such that MinimumFollowDistance and MaximumFollowDistance represents the safe operating range along the path and LagTime represents the delay, in seconds, that the follower should maintain from the leader. Additional values allow for more complex convoy configurations and are defined with respect to the path of the lead vehicle; the LateralOffset refers to the distance from the path in the ground plane, while

## Unclassified

VerticalOffset refers to the height above or below the path measured tangentially to the ground plane. Alternatively, the VerticalOffset value can be an absolute measure of the desired altitude with respect to Mean Sea Level (MSL), the Ground Level (AGL), or the Sea Floor (ASL). When not explicitly specified in the message, the offset values are assumed to be equal to the relative position of the follower with respect to the leader at the time the SetFollowerState( 'START' ) is received. Any specified offset values that cannot be satisfied due to physical constraints, such as Z-offset in a ground system or orientation in a fixed wing aircraft, shall be ignored. This message may also contain optional maximum error values for each offset dimension. When the follower error exceeds these bounds, the follower should stop and may also stop the lead vehicle if the appropriate flag is set in the ErrorBehavior bitfield. If the ALLOW\_LEADER\_OVERRIDE bit is set, the follower may attempt to slow down or speed up the lead vehicle prior to exceeding the error bounds. If the error values are not specified in the message, they are assumed to be infinite.

**Table 3.2-3: SET FOLLOWER CONFIGURATION MESSAGE ENCODING**

<div style="text-align: center;"> body  └─ <b>record</b> name = FollowerRec </div>					
<b>Record Name = FollowerRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<presence_vector>	unsigned short integer			
2	<bit_field> Leader_ID	unsigned integer	bit_field	true	Bits 0..7 : Component ID of Leader Bits 8..15 : Node ID of Leader Bits 16..31 : Subsystem ID of Leader
3	<bit_field> ErrorBehavior	unsigned byte	one	true	Bit 0: STOP_LEADER Bit 1: ALLOW_LEADER_OVERRIDE
4	<fixed_field> LagTime	unsigned integer	Second	true	(scaled range = [0,3600], round )
5	<fixed_field> MinimumFollowDistance	unsigned integer	meter	true	(scaled range = [0,100000], round )
6	<fixed_field> MaximumFollowDistance	unsigned integer	meter	true	(scaled range = [0,100000], round )
7	<fixed_field> LateralOffset	unsigned integer	meter	true	(scaled range = [-100000,100000], round )

## Unclassified

8	<fixed_field> MaxLateralError	unsigned integer	meter	true	(scaled range = [0,100000], round )
9	<fixed_field> VerticalOffset	unsigned integer	meter	true	(scaled range = [-100000,100000], round )
10	<fixed_field> MaxVerticalError	unsigned integer	meter	true	(scaled range = [0,100000], round )
11	<fixed_field> VerticalOffsetType	unsigned byte	one	true	Value set, offset=false, ranges/enums: 0= DEPTH_MSL 1= DEPTH_AGL 2= DEPTH_ASF 3= RELATIVE
12	<fixed_field> Roll	unsigned short integer	radian	true	(scaled range = [-PI,PI], round )
13	<fixed_field> Max_Roll_Error	unsigned short integer	radian	true	(scaled range = [0,2*PI], round )
14	<fixed_field> Pitch	unsigned short integer	radian	true	(scaled range = [-PI,PI], round )
15	<fixed_field> Max_Pitch_Error	unsigned short integer	radian	true	(scaled range = [0,2*PI], round )
16	<fixed_field> Heading	unsigned short integer	radian	true	(scaled range = [-PI,PI], round )
17	<fixed_field> Max_Heading_Error	unsigned short integer	radian	true	(scaled range = [0,2*PI], round )

### 3.2.4.1.3 ID FFF1: SetFollowerState

This message allows a service to start or stop following behavior.

**Table 3.2-4: SET FOLLOWER STATE MESSAGE ENCODING**

body └ record name = FollowerStateRec					
Record Name = FollowerStateRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> State	unsigned byte	one	false	Value set, offset=false, ranges/enums: 0= STOP 1= START

### 3.2.4.2 Output Set

#### 3.2.4.2.1 ID FFF4: ReportFollowerConfiguration

Table 3.2-5: REPORT FOLLOWER CONFIGURATION MESSAGE ENCODING

<pre> body ├── sequence name = FollowerSeq │   ├── record name = FollowerStateRec │   └── record name = FollowerRec </pre>					
<b>Record Name = FollowerStateRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> State	unsigned byte	one	false	Value set, offset=false, ranges/enums: 0= STOP 1= START
<b>Record Name = FollowerRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<presence_vector>	unsigned short integer			
2	<bit_field> Leader_ID	unsigned integer	bit_field	true	Bits 0..7 : Component ID of Leader Bits 8..15 : Node ID of Leader Bits 16..31 : Subsystem ID of Leader
3	<bit_field> ErrorBehavior	unsigned byte	one	true	Bit 0: STOP_LEADER Bit 1: ALLOW_LEADER _OVERRIDE
4	<fixed_field> LagTime	unsigned integer	Second	true	(scaled range = [0,3600], round )
5	<fixed_field> MinimumFollowDistance	unsigned integer	meter	true	(scaled range = [0,100000], round )
6	<fixed_field> MaximumFollowDistance	unsigned integer	meter	true	(scaled range = [0,100000], round )
7	<fixed_field> LateralOffset	unsigned integer	meter	true	(scaled range = [- 100000,100000], round )



## Unclassified

8	<b>&lt;fixed_field&gt;</b> MaxLateralError	unsigned integer	meter	true	(scaled range = [0,100000], round )
9	<b>&lt;fixed_field&gt;</b> VerticalOffset	unsigned integer	meter	true	(scaled range = [-100000,100000], round )
10	<b>&lt;fixed_field&gt;</b> MaxVerticalError	unsigned integer	meter	true	(scaled range = [0,100000], round )
11	<b>&lt;fixed_field&gt;</b> VerticalOffsetType	unsigned byte	one	true	Value set, offset=false, ranges/enums: 0= DEPTH_MSL 1= DEPTH_AGL 2= DEPTH_ASF 3= RELATIVE
12	<b>&lt;fixed_field&gt;</b> Roll	unsigned short integer	radian	true	(scaled range = [-PI,PI], round )
13	<b>&lt;fixed_field&gt;</b> Max_Roll_Error	unsigned short integer	radian	true	(scaled range = [0,2*PI], round )
14	<b>&lt;fixed_field&gt;</b> Pitch	unsigned short integer	radian	true	(scaled range = [-PI,PI], round )
15	<b>&lt;fixed_field&gt;</b> Max_Pitch_Error	unsigned short integer	radian	true	(scaled range = [0,2*PI], round )
16	<b>&lt;fixed_field&gt;</b> Heading	unsigned short integer	radian	true	(scaled range = [-PI,PI], round )
17	<b>&lt;fixed_field&gt;</b> Max_Heading_Error	unsigned short integer	radian	true	(scaled range = [0,2*PI], round )

### 3.2.5 Protocol Behavior

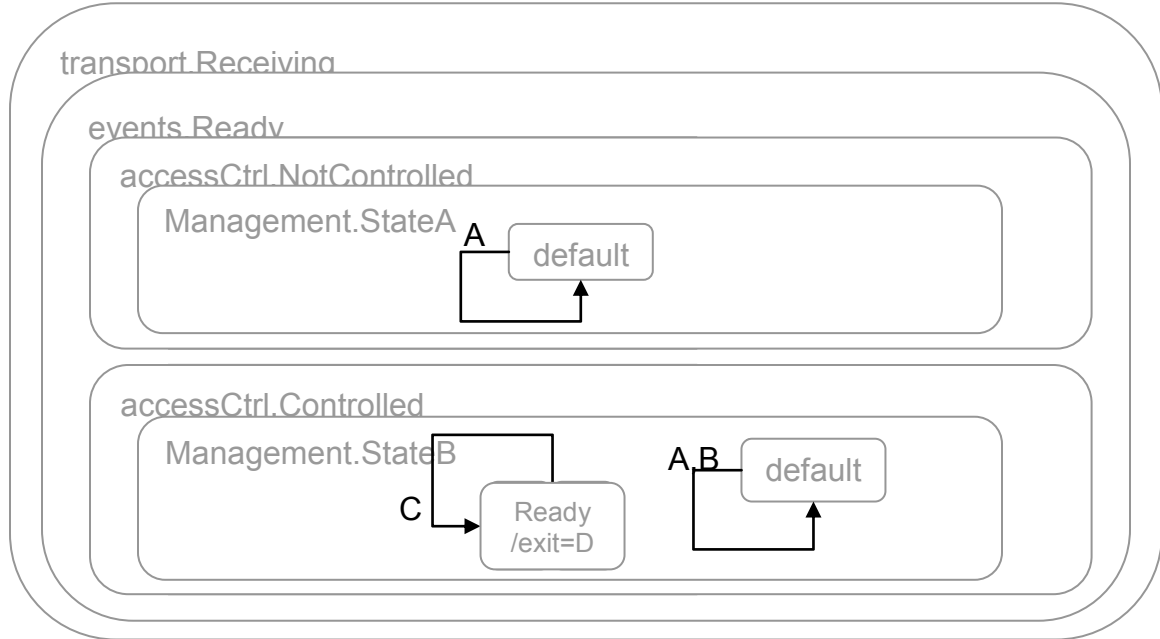


Figure 3.2-2: LEADER FOLLOWER DRIVER SERVICE PROTOCOL BEHAVIOR

Table 3.2-6: LEADER FOLLOWER DRIVER SERVICE EXIT STATE TRANSITIONS

Label	State	Type	Guard	Actions
D	Ready	Exit	isErrorBehaviorLeaderSt op	sendStopToLeader resetFollowerState
	Ready	Exit		resetFollowerState

Table 3.2-7: LEADER FOLLOWER DRIVER SERVICE STATE TRANSITIONS

Label	Trigger	Conditions	Actions
A	QueryFollowerConfiguration		SendResponse message 'ReportFollowerConfigura tion'
B	SetFollowerConfiguration	isControllingClient()	setFollowerValues ( msg )
C	SetFollowerState	isControllingClient()	setFollowerState ( msg )

Table 3.2-8: LEADER FOLLOWER DRIVER SERVICE CONDITIONS

Condition	Interpretation
isControllingClient	True is the command message was received from the

## Unclassified

client currently controlling this component

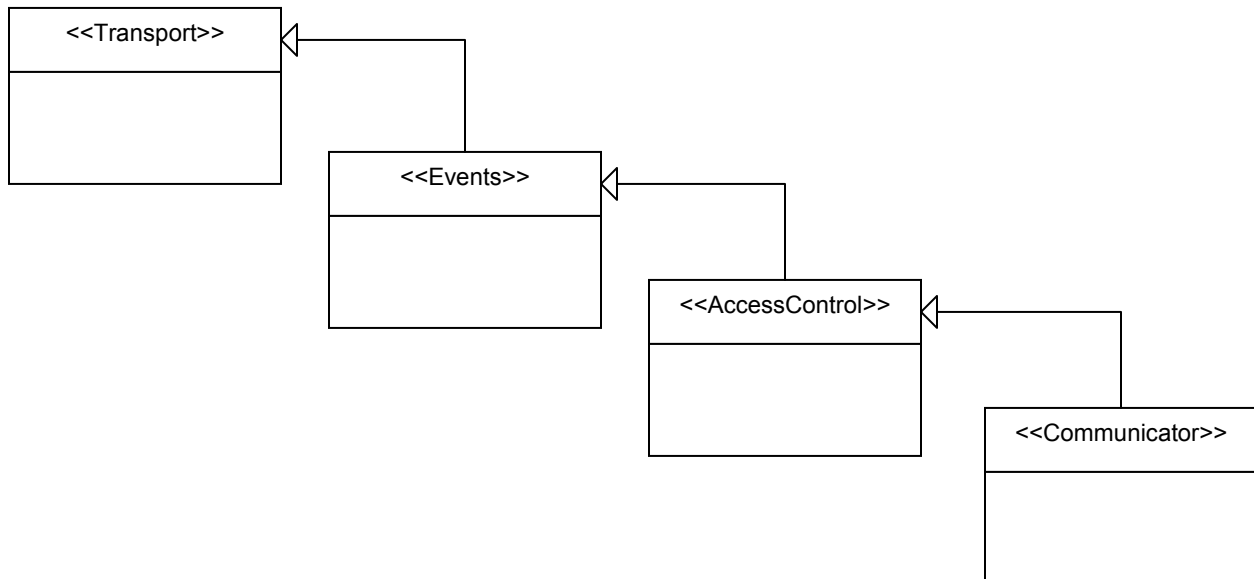
**Table 3.2-9: LEADER FOLLOWER DRIVER SERVICE TRANSITION ACTIONS**

Action	Interpretation
SendResponse	Send the specified response message
setFollowerValues	Set the specified configuration values
setFollowerState	Set the specified follower state
resetFollowerState	Set following state to STOP
sendStopToLeader	Send an E-STOP request to the leader

### 3.3 Communicator Service

name=Communicator  
version=0.1  
id=urn:jpo:comms:Communicator

Inherits-from AccessControl  
name=accessControl  
id= urn:jaus:jss:core:AccessControl  
version=1.0



**Figure 3.3-1: COMMUNICATOR SERVICE**

### 3.3.1 Description

The Communicator Service provides a mechanism for run-time configuration and monitoring of a communication link, usually a radio.

### 3.3.2 Assumptions

Messages may be delayed, lost or reordered. Each instance of the communicator service corresponds to a single communication link. Multiple communication links may be present on a single subsystem.

### 3.3.3 Vocabulary

**Table 3.3-1: COMMUNICATOR SERVICE VOCABULARY**

Message ID (hex)	Name	Command
<b>Input Set</b>		
2900	<a href="#">QueryCommunicatorCapability</a>	false
2901	<a href="#">QueryCommunicatorConfiguration</a>	false
2902	<a href="#">QueryCommunicatorHealth</a>	false
0901	<a href="#">SetCommunicatorConfiguration</a>	true
<b>Output Set</b>		
4900	<a href="#">ReportCommunicatorCapability</a>	false
4901	<a href="#">ReportCommunicatorConfiguration</a>	false
4902	<a href="#">ReportCommunicatorHealth</a>	false
0902	<a href="#">SetCommunicatorConfigurationResponse</a>	false

**Table 3.3-2: COMMUNICATOR SERVICE INTERNAL EVENT SET**

Name	Interpretation
<i>ValidationTimeout</i>	Occurs when communications are not re-established after a configuration change
<i>CommsEstablished</i>	Occurs when communications are successfully re-established after a configuration change

### 3.3.4 Encoding

#### 3.3.4.1 Input Set

##### 3.3.4.1.1 ID 2900: *QueryCommunicatorCapability*

This message is used to query capabilities associated with the communication device.

**Table 3.3-3: QUERY COMMUNICATOR CAPABILITY MESSAGE ENCODING**

body └ (empty)
-------------------

**3.3.4.1.2 ID 2901: QueryCommunicatorConfiguration**

This message is used to query the current configuration associated with the communication device.

**Table 3.3-4: QUERY COMMUNICATOR CONFIGURATION MESSAGE ENCODING**

body └ record name = QueryCommunicatorConfigurationRec					
Record Name = QueryCommunicatorConfigurationRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> PresenceVector	unsigned short integer	one	false	

**3.3.4.1.3 ID 2902: QueryCommunicatorHealth**

This message is used to query the current health and status associated with the communication device.

**Table 3.3-5: QUERY COMMUNICATOR HEALTH MESSAGE ENCODING**

body └ <b>record</b> name = QueryCommunicatorHealthRec					
Record Name = QueryCommunicatorHealthRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> PresenceVector	unsigned short integer	one	false	

**3.3.4.1.4 ID 0901: SetCommunicatorConfiguration**

The Set Communicator Configuration message allows the client component to set configuration values for the communication device.

**Table 3.3-6: SET COMMUNICATOR CONFIGURATION MESSAGE ENCODING**

body

└

sequence

name = SetCommunicatorConfigurationSeq

record

name = SetCommunicatorConfigurationRec

variant

name = TransmitPowerVariant

record

name = ContinuousAdjustableRec

record

name = HighLowPowerRec

record

name = AutomaticPowerControlRec

Record Name = SetCommunicatorConfigurationRec

Field #	Name	Type	Units	Optional	Interpretation
1	<presence_vector>	unsigned short integer			
2	<fixed_field>	unsigned byte	one	true	Value enum

## Unclassified

	TransmitterStatus				0= TransmitOFF 1= TransmitON 2= Standby
3	<fixed_field> ModeOfOperation	unsigned byte	one	true	Value enum 0= Point_to_Point 1= Multipoint
4	<bit_field> EncryptionMode	Unsigned byte	One	True	Bit 0: TransmitterMode. When this bit is high, the transmitted data will be encrypted using AES 128.  Bit 1-2: ReceiverMode. The following enumeration shall apply: Enum 0: Receive encrypted only Enum 1: Receive non-encrypted only Enum 2: Receive both  Bit 3: Encryption type. When this bit is high, bulk (transec) encryption is used. Otherwise, packet (payload) encryption is used.
5	<variable_length_string> EncryptionKey	Count_field = unsigned byte	One	True	Public key for AES 128 encryption
6	<fixed_field> Channel	unsigned byte	one	true	
7	<fixed_field> FrequencyBand	Unsigned byte	One	true	Band index as defined by the Report Capabilities message.
8	<fixed_field> ModulationScheme	unsigned byte	one	true	Enum value:  0: Adaptive Modulation 1: 2-QAM 2: 2-PSK 3: 2-FSK 4: 4-QAM 5: 4-PSK 6: 4-FSK 7: 8-QAM 8: 8-PSK 9: 8-FSK 10: 16-QAM 11: 16-PSK 12: 16-FSK 13: 32-QAM 14: 32-PSK 15: 32-FSK 16: 64-QAM 17: 64-PSK 18: 64-FSK
9	<fixed_field> OccupiedBandwidth	unsigned integer	Hertz	true	Range of 0 to 100 Mhz scaled range = [0,100000000]
<b>Record Name</b> = ContinuousAdjustableRec					
<b>Field #</b>	<b>Name</b>	<b>Type</b>	<b>Units</b>	<b>Optional</b>	<b>Interpretation</b>
1	<fixed_field> TransmitPower	unsigned short integer	watt	False	scaled range = [0,10]

## Unclassified

Record Name = HighLowPowerRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> TransmitPower	unsigned byte	one	False	Enum values: LOW_POWER HIGH_POWER
Record Name = AutomaticPowerControlRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> MinTransmitPower	unsigned short integer	watt	False	scaled range = [0,10]
2	<fixed_field> MaxTransmitPower	unsigned short integer	watt	False	scaled range = [0,10]

### 3.3.4.2 Output Set

#### 3.3.4.2.1 ID 4900: ReportCommunicatorCapability

This message is used to report the capabilities associated with this communication device.

**Table 3.3-7: REPORT COMMUNICATOR CAPABILITY MESSAGE ENCODING**

<div style="margin-left: 40px;"> body <div style="margin-left: 20px;"> sequence name = ReportCommunicatorCapabilitiesSeq record name = ReportCommunicatorCapabilityRec list name = FrequencyBandList (count_field = unsigned byte) record name = FrequencyBandRec list name = ChannelList (count_field = unsigned byte) record name = ChannelRec </div> </div>					
Record Name = ReportCommunicatorCapabilityRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<presence_vector>	unsigned short integer			
2	<bit_field> SupportedTransmitterStatus	unsigned byte	bit_field	true	For each bit, a value of zero means the status is not supported by the comms device.  Bit 0: Transmit ON Bit 1: Transmit OFF Bit 2: Standby
3	<bit_field> SupportedModesOfOperation	unsigned byte	bit_field	true	For each bit, a value of zero means the mode is not supported by the comms device.  Bit 0: Point to Point Bit 1: Multipoint
4	<bit_field> SupportedEncryptionModes	Unsigned byte	One	True	For each bit, a value of zero means the mode is not supported by the comms device.  Bit 0: Transmitter

## Unclassified

					<p>supports AES 128 encryption.</p> <p>Bit 1: Receiver supports encrypted-only mode            Bit 2: Receiver supports non-encrypted only            Bit 3: Receiver supports promiscuous mode</p> <p>Bit 4: Bulk/Package mode. If this bit is set, encryption is bulk (transec). Otherwise, encryption is packet (payload).</p>
5	<b>&lt;fixed_field&gt;</b> MinTransmitPower	unsigned short integer	watt	true	scaled range = [0,10]
6	<b>&lt;fixed_field&gt;</b> MaxTransmitPower	unsigned short integer	watt	true	scaled range = [0,10]
7	<b>&lt;bit_field&gt;</b> SupportedTransmitPowerLevels	Unsigned byte	One	True	<p>For each bit, a value of zero means the power level is not supported by the comms device.</p> <p>Bit 0: Continuous Scale Adjustable            Bit 1: Low Power            Bit 2: High Power            Bit 3: Automatic Transmit Power Control</p>
8	<b>&lt;bit_field&gt;</b> SupportedModulationSchemes	unsigned integer	one	true	<p>For each bit, a value of zero means the modulation scheme is not supported:</p> <p>Bit 0: Adaptive Modulation            Bit 1: 2-QAM            Bit 2: 2-PSK            Bit 3: 2-FSK            Bit 4: 4-QAM            Bit 5: 4-PSK            Bit 6: 4-FSK            Bit 7: 8-QAM            Bit 8: 8-PSK            Bit 9: 8-FSK            Bit 10: 16-QAM            Bit 11: 16-PSK            Bit 12: 16-FSK            Bit 13: 32-QAM            Bit 14: 32-PSK            Bit 15: 32-FSK            Bit 16: 64-QAM            Bit 17: 64-PSK            Bit 18: 64-FSK</p>
9	<b>&lt;fixed_field&gt;</b> MinimumOccupiedBandwidth	unsigned integer	Hertz	true	Range of 0 to 100 Mhz scaled range = [0,100000000]
10	<b>&lt;fixed_field&gt;</b> MaximumOccupiedBandwidth	unsigned integer	Hertz	true	Range of 0 to 100 Mhz scaled range = [0,100000000]
<b>Record Name</b> = FrequencyBandList					
1	<b>&lt;fixed_field&gt;</b>	unsigned byte	one	false	Defines a band for multi-



## Unclassified

	BandIndex				band radios. Each band is defined in the range of [Min, Max].
2	<fixed_field> MinFrequency	unsigned long integer	Hertz	false	Range of 100 kHz to 10 GHz scaled range = [100000,100000000000]
3	<fixed_field> MaxFrequency	unsigned long integer	Hertz	false	Range of 100 kHz to 10 GHz scaled range = [100000,100000000000]
<b>Record Name = ChannelRec</b>					
1	<fixed_field> ChannelNumber	unsigned byte	one	false	For multi-band radios, channel numbers may be repeated, but each duplicate channel number must have a unique band index.
2	<fixed_field> CenterFrequency	unsigned long integer	Hertz	false	Range of 100 kHz to 10 GHz scaled range = [100000,100000000000]
3	<fixed_field> FrequencyBand	Unsigned byte	One	false	Band index

### 3.3.4.2.2 ID 4901: ReportCommunicatorConfiguration

This message is used to report the current configuration associated with this communication device.

**Table 3.3-8: REPORT COMMUNICATOR CONFIGURATION MESSAGE ENCODING**

<div style="text-align: center;"> body  └─ <b>sequence</b> name = ReportCommunicatorConfigurationSeq  <b>record</b> name = ReportCommunicatorConfigurationRec  <b>variant</b> name = TransmitPowerVariant  <b>record</b> name = ContinuousAdjustableRec  <b>record</b> name = HighLowPowerRec  <b>record</b> name = AutomaticPowerControlRec </div>					
<b>Record Name = ReportCommunicatorConfigurationRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<presence_vector>	unsigned short integer			
2	<fixed_field> TransmitterStatus	unsigned byte	one	true	Value enum 0= TransmitOFF 1= TransmitON 2= Standby
3	<fixed_field> ModeOfOperation	unsigned byte	one	true	Value enum 0= Point_to_Point 1= Multipoint
4	<bit_field> EncryptionMode	Unsigned byte	One	True	Bit 0: TransmitterMode. When this bit is high, the transmitted data will be encrypted using AES 128.  Bit 1-2: ReceiverMode.

## Unclassified

					<p>The following enumeration shall apply:</p> <p>Enum 0: Receive encrypted only</p> <p>Enum 1: Receive non-encrypted only</p> <p>Enum 2: Receive both</p> <p>Bit 3: Encryption type. When this bit is high, bulk (transec) encryption is used. Otherwise, packet (payload) encryption is used.</p>
5	<fixed_field> Channel	unsigned byte	one	true	
6	<fixed_field> FrequencyBand	Unsigned byte	One	true	Band index as defined by the Report Capabilities message.
7	<bit_field> ActiveModulationScheme	unsigned byte	one	true	<p>Bit 0: Adaptive Modulation Active (when this bit is set, adaptive modulation is active. The instantaneous modulation type is given by bits 1-7).</p> <p>Bits 1-7: Current (instantaneous) modulation type based on the following enumeration:</p> <p>1: 2-QAM 2: 2-PSK 3: 2-FSK 4: 4-QAM 5: 4-PSK 6: 4-FSK 7: 8-QAM 8: 8-PSK 9: 8-FSK 10: 16-QAM 11: 16-PSK 12: 16-FSK 13: 32-QAM 14: 32-PSK 15: 32-FSK 16: 64-QAM 17: 64-PSK 18: 64-FSK</p>
8	<fixed_field> OccupiedBandwidth	unsigned integer	Hertz	true	Range of 0 to 100 Mhz scaled range = [0,100000000]
<b>Record Name = ContinuousAdjustableRec</b>					
<b>Field #</b>	<b>Name</b>	<b>Type</b>	<b>Units</b>	<b>Optional</b>	<b>Interpretation</b>
1	<fixed_field> TransmitPower	unsigned short integer	watt	False	scaled range = [0,10]
<b>Record Name = HighLowPowerRec</b>					
<b>Field #</b>	<b>Name</b>	<b>Type</b>	<b>Units</b>	<b>Optional</b>	<b>Interpretation</b>
1	<fixed_field> TransmitPower	unsigned byte	one	False	Enum values: LOW_POWER HIGH_POWER

## Unclassified

Record Name = AutomaticPowerControlRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> MinTransmitPower	unsigned short integer	watt	False	scaled range = [0,10]
2	<fixed_field> MaxTransmitPower	unsigned short integer	watt	False	scaled range = [0,10]

### 3.3.4.2.3 ID 4902: ReportCommunicatorHealth

This message is used to report the current health and status associated with this communication device.

**Table 3.3-9: REPORT COMMUNICATOR HEALTH MESSAGE ENCODING**

<div style="margin-left: 40px;"> body  └─ <b>sequence</b> name = ReportCommunicatorHealthSeq  <b>record</b> name = ReportCommunicatorHealthRec  <b>variant</b> name = ErrorVariant            <b>record</b> name = PacketErrorCountRec            <b>record</b> name = PacketErrorRateRec            <b>record</b> name = BitErrorCountRec            <b>record</b> name = BitErrorRateRec  <b>list</b> name = ChannelList            (count_field = unsigned byte)            <b>record</b> name = ChannelRec </div>					
Record Name = ReportCommunicatorHealthRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<presence_vector>	unsigned byte			
2	<fixed_field> BIT_Results	unsigned byte	one	true	Enumeration values: 0= Passed 1= Failed
3	<fixed_field> Latency	unsigned short integer	second	true	scaled range = [0,1]
4	<fixed_field> DataRate	unsigned short integer	one	true	Measured in Mbps scaled range = [0,1000]
5	<fixed_field> ReceivedSignalPower	unsigned short integer	one	true	Measured in dBm scaled range = [-120,-20]
6	<fixed_field> ErrorVectorMagnitude	unsigned short integer	one	true	Percent scaled range = [0,100]
Record Name = PacketErrorCountRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> PacketErrorCount	Unsigned integer	one	false	Number of packet errors in the last one second
Record Name = PacketErrorRateRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> PacketErrorRate	Unsigned integer	one	false	Packet errors as a percentage of total packets sent Scale range = [0, 100]
Record Name = BitErrorCountRec					

## Unclassified

Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> BitErrorCount	Unsigned integer	one	false	Number of bit errors in the last one second
<b>Record Name = BitErrorRateRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> BitErrorRate	Unsigned integer	one	false	Bit errors as a percentage of total bits sent Scale range = [0, 100]
<b>Record Name = SNR_Per_Channel_Rec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Channel	unsigned byte	one	false	
2	<fixed_field> SNR	unsigned short integer	one	false	scaled range = [0,100]

### 3.3.4.2.4 ID 0902: SetCommunicatorConfigurationResponse

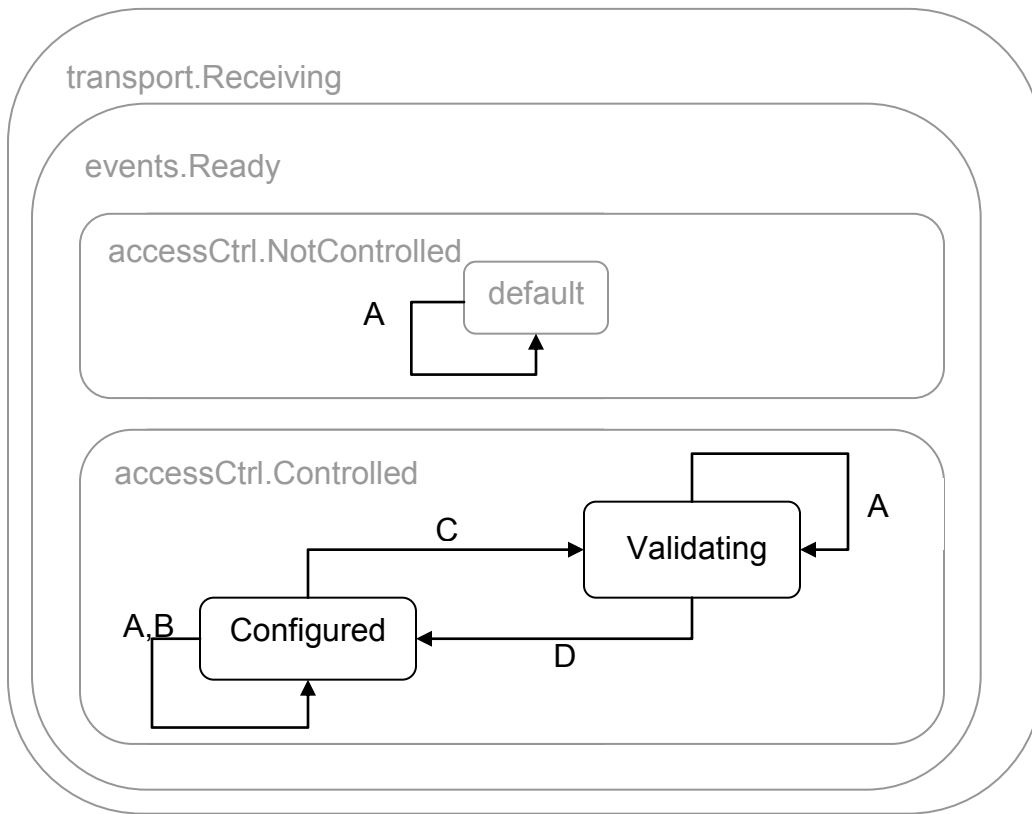
The Set Communicator Configuration Response message is sent as a notification back to a client on the status of the SetCommunicatorConfiguration message.

**Table 3.3-10: SET COMMUNICATOR CONFIGURATION RESPONSE MESSAGE ENCODING**

<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">body</div> <div style="border-left: 1px solid black; padding-left: 10px;"> <b>record</b> name = SetCommunicatorConfigurationResponseRec </div> </div>					
<b>Record Name = SetCommunicatorConfigurationResponseRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
	<bit_field> Result	unsigned byte	bit_field	false	For each bit, a value of 1 means the command value was invalid  Bit 0: TransmitterStatusInvalid Bit 1: ModelInvalid Bit 2: ChannelInvalid Bit 3: PowerLevelInvalid Bit 4: ModulationInvalid Bit 5: BandwidthInvalid Bit 6: EncryptionModelInvalid

### 3.3.5 Protocol Behavior

The Communicator uses a 'safety valve' mechanism to provide fault-tolerant changes to configuration; when a configuration change is requested, communications must be re-established within a specified period of time or the previous configuration is restored (reverted).



**Figure 3.3-2: COMMUNICATOR SERVICE PROTOCOL BEHAVIOR**

**Table 3.3-11: COMMUNICATOR SERVICE STATE TRANSITIONS**

Label	Trigger	Conditions	Actions
A	QueryCommunicatorCapability		SendResponse message 'ReportCommunicatorCapability'
	QueryCommunicatorConfiguration		SendResponse message 'ReportCommunicatorConfiguration'
	QueryCommunicatorHealth		SendResponse message 'ReportCommunicatorHealth'
B	SetCommunicatorConfiguration	isControllingClient() && ! isValidCommand()	SendResponse message 'SetCommunicatorResponse'
C	SetCommunicatorConfiguration	isControllingClient() && isValidCommand()	storeCurrentConfigurationValues() setConfigurationValues ( msg ) SendResponse message 'SetCommunicatorResponse'

## Unclassified

			e'
D	QueryCommunicatorCapability	isControllingClient()	SendResponse message 'ReportCommunicatorCapability'
	QueryCommunicatorConfiguration	isControllingClient()	SendResponse message 'ReportCommunicatorConfiguration'
	QueryCommunicatorHealth	isControllingClient()	SendResponse message 'ReportCommunicatorHealth'
	<i>ValidationTimeout</i>		revertConfigurationValues()
	<i>CommsEstablished</i>		

**Table 3.3-12: COMMUNICATOR SERVICE CONDITIONS**

Condition	Interpretation
isValidCommand	True if all command values specified in the message are valid
isControllingClient	True is the command message was received from the client currently controlling this component

**Table 3.3-13: COMMUNICATOR SERVICE TRANSITION ACTIONS**

Action	Interpretation
SendResponse	Send the specified response message
SetConfigurationValues	Set the specified configuration values
storeCurrentConfigurationValues	Store the current configuration values prior to setting the new values, in case the configuration needs to be reverted
revertConfigurationValues	Revert to the previously stored configuration

### 3.4 Platform Mode Service

name=PlatformMode

version=0.1

id=urn:jpo:platformmanager:PlatformMode

Inherits-from AccessControl

name=AccessControl

id= urn:jaus:jss:core:AccessControl

version=1.0

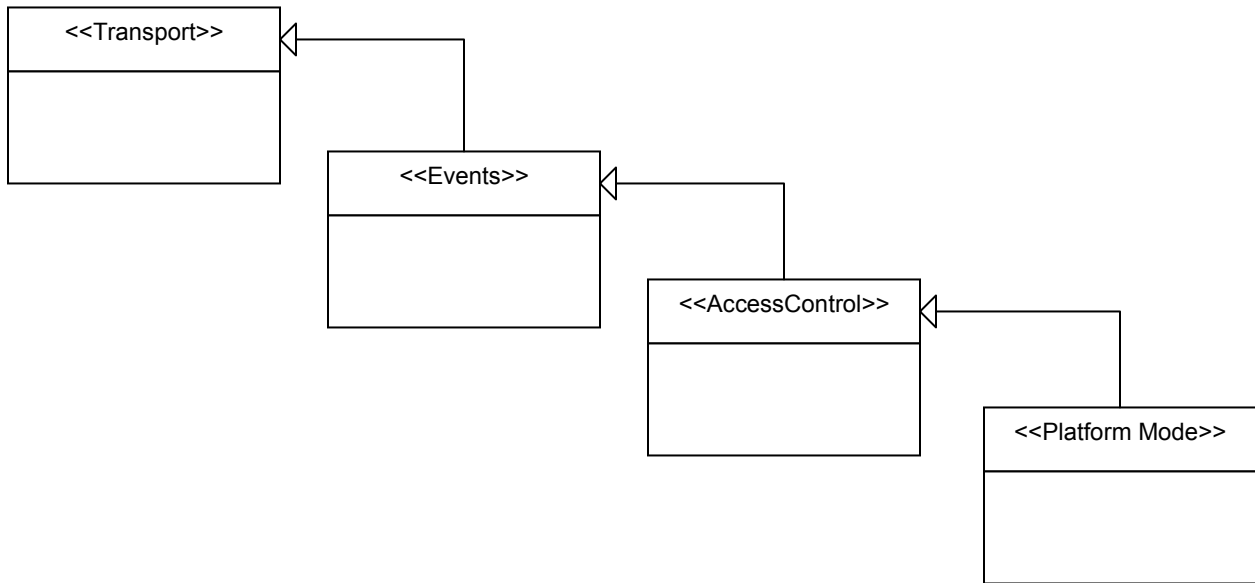


Figure 3.4-1: PLATFORM MODE SERVICE

### 3.4.1 Description

The Platform Mode Service provides a mechanism for informing of a mode change.

### 3.4.2 Assumptions

Messages may be delayed, lost or reordered.

### 3.4.3 Vocabulary

Table 3.4-1: PLATFORM MODE SERVICE VOCABULARY

<i>Message ID (hex)</i>	<i>Name</i>	<i>Command</i>
<b>Input Set</b>		
FF20	<a href="#"><i>QueryPlatformMode</i></a>	false
FF21	<a href="#"><i>QueryAvailableModes</i></a>	false
FF22	<a href="#"><i>SetPlatformMode</i></a>	true
<b>Output Set</b>		
FF23	<a href="#"><i>ReportPlatformMode</i></a>	false
FF24	<a href="#"><i>ReportAvailableModes</i></a>	false

### 3.4.4 Encoding

#### 3.4.4.1 Input Set

##### 3.4.4.1.1 ID FF20 *QueryPlatformMode*

This message is used to query the current platform mode.

**Table 3.4-2: QUERY PLATFORM MODE MESSAGE ENCODING**

body └ (empty)
-------------------

#### **3.4.4.1.2 ID FF21 QueryAvailableModes**

This message is used to query the supported available modes.

**Table 3.4-3: QUERY AVAILABLE MODES MESSAGE ENCODING**

body └ (empty)
-------------------

#### **3.4.4.1.3 ID FF22 SetPlatformMode**

This message is used to set the current platform mode of a component.

**Table 3.4-4: SET PLATFORM MODE MESSAGE ENCODING**

body └ <b>record</b> name = SetPlatformModeRec					
<b>Record Name = SetPlatformModeRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Mode	unsigned byte	one	false	Enumerated Values 0 = Operational
2	<fixed_field> DriveSubMode	unsigned byte	One	False	Enumerated values  0 = RC Teleop 1 = Waypoint Navigation 2 = Leader Follower

### **3.4.4.2 Output Set**

#### **3.4.4.2.1 ID FF23: ReportPlatformMode**

This message is used to report the current platform mode.

**Table 3.4-5: REPORT PLATFORM MODE MESSAGE ENCODING**

body └ <b>record</b> name = ReportPlatformModeRec
--



## Unclassified

**Record Name** = ReportPlatformModeRec

Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Mode	unsigned byte	one	false	Enumerated Values 0 = None 1 = Operational
2	<fixed_field> DriveSubMode	unsigned byte	One	false	Enumerated values  0 = RC Teleop 1 = Waypoint Navigation 2 = Leader Follower

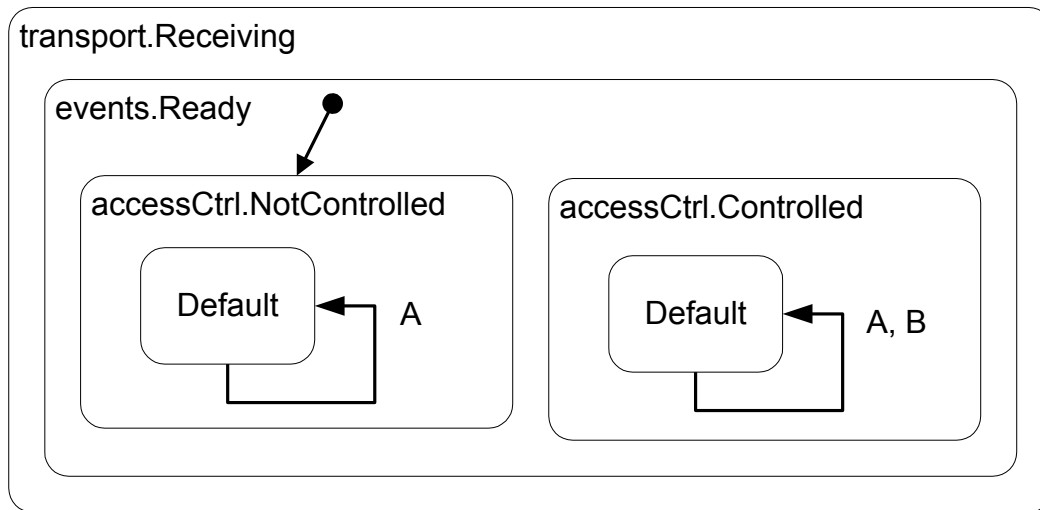
### 3.4.4.2.2 ID FF24: ReportAvailableModes

This message is used to report the supported modes.

**Table 3.4-6: REPORT AVAILABLE MODES MESSAGE ENCODING**

<div style="text-align: center;"> body  └─ list name = ModeList  (count_field = unsigned byte)  record name = ModeRec </div>					
<b>Record Name</b> = ModeRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Mode	unsigned byte	one	false	Enumerated Values 0 = Operational
2	<bit_field> DriveSubModes	unsigned byte	bit_field	false	For each bit, a value of 1 means the mode is available  Bit 0: RC Teleop Bit 1: Waypoint Navigation Bit 2: Leader Follower

### 3.4.5 Protocol Behavior



**Figure 3.4-2: PLATFORM MODE SERVICE PROTOCOL BEHAVIOR**

**Table 3.4-7: PLATFORM MODE SERVICE STATE TRANSITIONS**

Label	Trigger	Conditions	Actions
A	QueryPlatformMode		SendResponse message 'ReportPlatformMode'
	QueryAvailableModes		SendResponse message 'ReportAvailableModes'
B	SetPlatformMode	isControllingClient()	setPlatformMode( msg ) SendResponse 'ReportPlatformMode'

**Table 3.4-8: PLATFORM MODE SERVICE CONDITIONS**

Condition	Interpretation
isControllingClient	True if the command message was received from the client currently controlling this component

**Table 3.4-9: PLATFORM MODE SERVICE TRANSITION ACTIONS**

Action	Interpretation
SendResponse	Send the specified response message

## 3.5 Health Monitor Service

Name=HealthMonitor

Version = 0.1

Id=urn:jpo:health:HealthMonitor

Inherits-from: urn:jaus:jss:core:Events, v1.0

## Unclassified

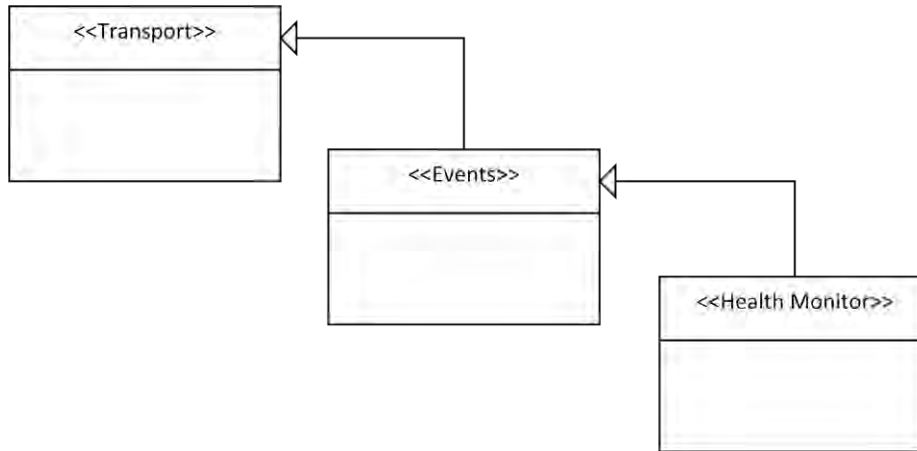


Figure 3.5-1: HEALTH MONITOR SERVICE

### 3.5.1 Description

The Health Monitor service indicates components and/or services that are in an emergency, failure, degraded, or comms lost state.

### 3.5.2 Assumptions

Messages may be delayed, lost, or reordered.

### 3.5.3 Vocabulary

Table 3.5-1: HEALTH MONITOR SERVICE VOCABULARY

Message ID (hex)	Name	Command
<b>Input Set</b>		
FF10	QueryComponentHealth	False
FF11	QueryServiceHealth	false
<b>Output Set</b>		
FF12	ReportComponentHealth	false
FF13	ReportServiceHealth	false

### 3.5.4 Encoding

#### 3.5.4.1 Input Message Set

##### 3.5.4.1.1 ID FF10: QueryComponentHealth

Allows consumers of the Health Monitor service to determine what components residing on a subsystem are in a failure, emergency, degraded, or comms lost state. This message shall request the health status summary of a subsystem or node or a single component. This message is closely related to the Query Service List message.

Table 3.5-2: QUERY COMPONENT HEALTH MESSAGE ENCODING

Body L <b>list</b> name=SubsystemList (count_field = unsigned short integer) <b>sequence</b> name=SubsystemSeq <b>record</b> name=SubsystemRec <b>list</b> name=NodeList (count_field = unsigned byte) <b>sequence</b> name=NodeSeq <b>record</b> name=NodeRec <b>list</b> name = ComponentList (count_field = unsigned byte) <b>record</b> name = ComponentRec					
<b>record</b> name=SubsystemRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Subsystem ID	unsigned short integer	one	false	Use 65535 (0xFFFF) if health status information from all subsystems in the system is required. If 0 is used, then the health status applies to the parent subsystem of the Health Monitor service.  Value_range 0: Health status for parent subsystem. 1-65534: valid Subsystem IDs 65535: All subsystems in the system.
<b>record</b> name=NodeRec					
1	<fixed_field> Node ID	unsigned byte	one	false	Use 255 if health status information from all nodes in the subsystem is required.  Value_range 0: Reserved 1-254: valid Node IDs 255: All nodes in the subsystem
<b>record</b> name=ComponentRec					
1	<fixed_field> Component ID	unsigned byte	one	false	Use 255 health status information from all components in the node are required.  Value_range 0: Reserved 1-254: valid component IDs 255: All components in the node.

#### 3.5.4.1.2 ID FF11: QueryServiceHealth

Allows consumers of the Health Monitor service to get a full list of what services are reporting errors or other significant health issues. This message request the health status for services based on subsystem, node, component, and an optional search filter.

Table 3.5-3: QUERY SERVICE HEALTH MESSAGE ENCODING

Body L <b>list</b> name=SubsystemList (count_field = unsigned short integer) <b>sequence</b> name=SubsystemSeq <b>record</b> name=SubsystemRec					
--	--	--	--	--	--

## Unclassified

<b>list</b> name=NodeList (count_field = unsigned byte) <b>sequence</b> name=NodeSeq <b>record</b> name=NodeRec <b>list</b> name = ComponentList (count_field = unsigned byte) <b>record</b> name = ComponentRec					
<b>record</b> name=SubsystemRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Subsystem ID	unsigned short integer	one	false	Use 65535 (0xFFFF) if health status information from all subsystems in the system is required. If 0 is used, then the health status applies to the parent subsystem of the Health Monitor service.  Value_range 0: Health status for parent subsystem. 1-65534: valid Node IDs 65535: All subsystems in the system.
<b>record</b> name=NodeRec					
1	<fixed_field> Node ID	unsigned byte	one	false	Use 255 if health status information from all nodes in the subsystem is required.  Value_range 0: Reserved 1-254: valid Node IDs 255: All nodes in the subsystem
<b>record</b> name=ComponentRec					
1	<presence_vector>	Unsigned byte	One	False	
2	<fixed_field> Component ID	unsigned byte	one	false	Use 255 health status information from all components in the node are required.  Value_range 0: Reserved 1-254: valid component IDs 255: All components in the node.
3	<variable_length_string> SearchFilter	Count_field = unsigned byte	One	True	An optional filter to apply to the search results. Only service identifiers that contain this string should be returned.

### 3.5.4.2 Output Message Set

#### 3.5.4.2.1 ID FF12: ReportComponentHealth

Returns a structure containing the component health information for subsystem, nodes, and components specified.

**Table 3.5-4: REPORT COMPONENT HEALTH MESSAGE ENCODING**

body L <b>list</b> name = SubsystemList (count_field = unsigned short integer) <b>sequence</b> name=SubsystemSeq <b>record</b> name=SubsystemRec <b>list</b> name = NodeList
---

## Unclassified

(count_field = unsigned byte) L <b>sequence</b> name = NodeSeq L <b>record</b> name = NodeRec L <b>list</b> name = ComponentList (count_field = unsigned byte) L <b>record</b> name = ComponentHealthRec					
<b>record</b> name=SubsystemRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> SubsystemID	Unsigned short integer	One	False	Subsystem ID.
<b>record</b> name=NodeRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> NodeID	Unsigned byte	One	False	Node ID.
<b>record</b> name=ComponentHealthRec					
1	<fixed_field> ComponentID	Unsigned integer	One	False	Component ID. The full ID of the component that health is being reported for, where the least significant byte is the component ID, the next least significant byte is the node ID, and the highest two bytes are the subsystem ID.
2	<fixed_field> HealthState	Unsigned short integer	One	False	0: Failure 1: Emergency 2: Degraded 3: Comms Lost
3	<variable_length_string> HealthMessage	Count_field= unsigned byte	One	False	Descriptive message about the health state.

### 3.5.4.2.2 ID FF13: ReportServiceHealth

Returns a list of ReportHealthRec containing the component ID, health state enumeration, and descriptive string of all the components on the subsystem that are in a failure, emergency, degraded, or comms lost state.

**Table 3.5-5: REPORT SERVICE HEALTH MESSAGE ENCODING**

Body L <b>list</b> name = SubsystemList (count_field = unsigned short integer) <b>sequence</b> name=SubsystemSeq <b>record</b> name=SubsystemRec L <b>list</b> name = NodeList (count_field = unsigned byte) L <b>sequence</b> name = NodeSeq L <b>record</b> name = NodeRec L <b>list</b> name = ComponentList (count_field = unsigned byte) L <b>sequence</b> name = ComponentSeq L <b>record</b> name = ComponentRec L <b>list</b> name = ServiceHealthList (count_field = unsigned byte)	
--	--

# Unclassified

L record name = ServiceHealthStatusRec					
record name=SubsystemRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> SubsystemID	Unsigned short integer	One	False	Subsystem ID.
record name=NodeRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> NodeID	Unsigned byte	One	False	Node ID.
record name=ComponentRec					
1	<fixed_field> ComponentID	Unsigned byte	One	False	Component ID.
record name=ServiceHealthStatusRec					
1	<variable_length_string> Service_urn	Count_field = unsigned byte	One	False	Urn of the service health status is being reported for.
2	<fixed_field> Severity	Unsigned byte	One	False	Value set, offset=false, ranges/enums: 0=NONE 1=INFO 2=WARN 3=ERR 4=FAIL [5,255]=<reserved>
3	<fixed_field> Code	Unsigned byte	One	False	Value set, offset=false, ranges/enums: 1=PROCESSOR 2=RAM 3=ROM 4=FILESYS 5=POWER 6=SENSOR 7=ACTUATOR 8=SOFT [9,254]=RESERVED 255=BUSY
4	<bit_field> Logtime	Unsigned integer	Bit_field	False	Bits 0..9, Value set, offset=false, ranges/enums:[0,999] Bits 10..15, Value set, offset=false, ranges/enums:[0,59] Bits 16..21, Value set, offset=false, ranges/enums:[0,59] Bits 22..26, Value set, offset=false, ranges/enums:[0:23] Bits 27..31, Value set, offset=false, Ranges/enums[1,31]
5	<variable_length_string> Descriptor	Variable length string (byte[])	N/A	False	(Length min..max = 0..32)

### 3.5.5 Protocol Behavior

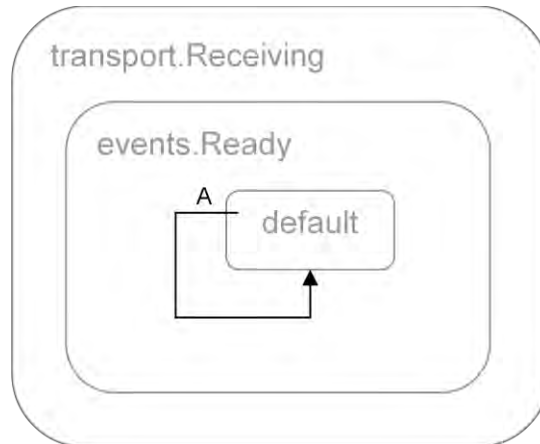


Figure 3.5-2: HEALTH MONITOR SERVICE PROTOCOL BEHAVIOR

Table 3.5-6: HEALTH MONITOR SERVICE STATE TRANSITIONS

Label	Trigger	Conditions	Actions
A	QueryComponentHealth		sendReportComponentHealth
A	QueryServiceHealth		sendReportServiceHealth

Table 3.5-7: HEALTH MONITOR SERVICE TRANSITION ACTIONS

Action	Interpretation
sendReportComponentHealth	Sends a ReportComponentHealth message to the requesting client.
sendReportServiceHealth	Sends a ReportServiceHealth message to the requesting client.

## 3.6 Health Reporter Service

Name=HealthReporter

Version = 0.1

Id=urn:jpo:health:HealthReporter

Inherits-from: urn:jaus:jss:core:AccessControl, v1.0



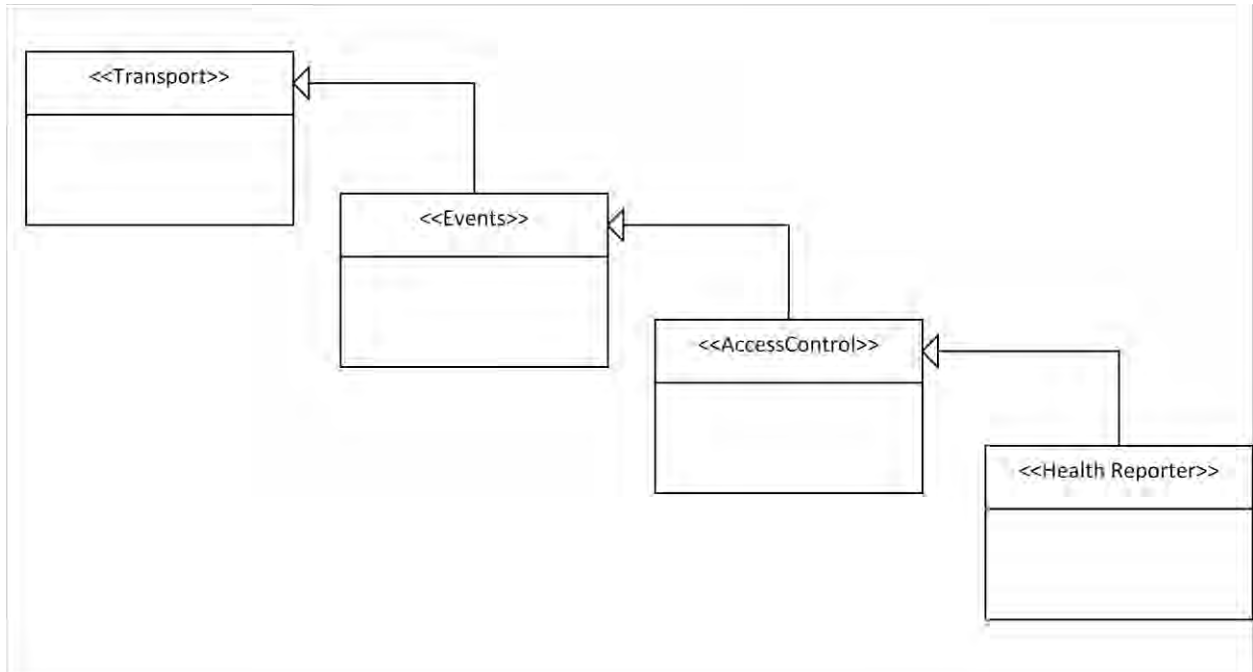


Figure 3.6-1: HEALTH REPORTER SERVICE

### 3.6.1 Description

The Health Reporter attributes defines a Health Reporter service that is used to perform built-in test (BIT) operations at Power-On and subsets of built-in-test in the background during runtime (RBIT) and when requested via command message (CBIT). The Health Reporter service maintains a record of the most current BIT results, and provides the most current BIT results when requested by a client or configured event.

### 3.6.2 Assumptions

Messages may be delayed, lost, or reordered.

### 3.6.3 Vocabulary

Table 3.6-1: HEALTH REPORTER SERVICE VOCABULARY

Message ID (hex)	Name	Command
<b>Input Set</b>		
ED01	QueryHealthStatus	False
DD01	PerformCBIT	True
<b>Output Set</b>		
FD01	ReportHealthStatus	false

### 3.6.4 Encoding

#### 3.6.4.1 Input Message Set

##### 3.6.4.1.1 ID ED01: QueryHealthStatus

The QueryHealthStatus message is used to request the most recent status from the Health Reporter.

**Table 3.6-2: QUERY HEALTH STATUS MESSAGE ENCODING**

body L (empty)
empty message body

##### 3.6.4.1.2 ID DD01: PerformCBIT

This message is used to request Health Reporter to initiate execution of Commanded BIT (CBIT).

**Table 3.6-3: PERFORM CBIT MESSAGE ENCODING**

body L (empty)
empty message body

#### 3.6.4.2 Output Message Set

##### 3.6.4.2.1 ID FD01: ReportHealthStatus

This message provides a report of the most recent status. The Report is constituted as a List, so that multiple status records may be reported in a single message; this enables a module to report errors from multiple services, and multiple errors per service. Errors not unique to a given service (as some hardware test failures) are reported with the service\_urn of the Health Reporter service.

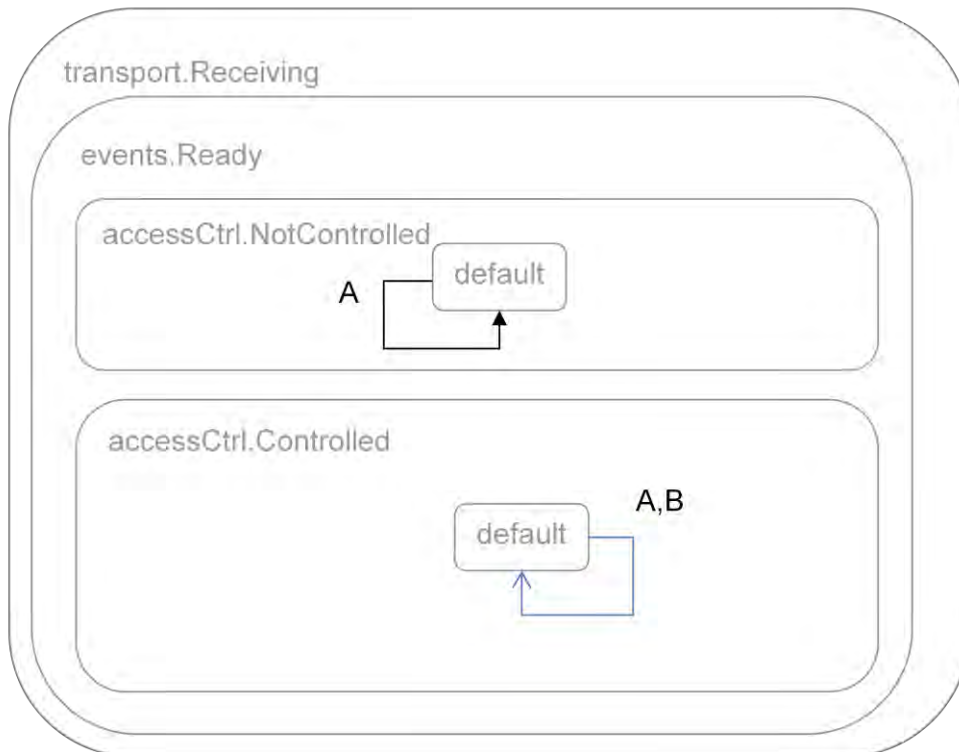
**Table 3.6-4: REPORT HEALTH STATUS MESSAGE ENCODING**

Body L list name = ComponentStatus (count_field = unsigned byte) record name=status					
record name=Status					
Field #	Name	Type	Units	Optional	Interpretation
1	<variable_length_string> service_urn	Variable length string (byte[])	One	False	(Length min..max = 0..64)
2	<fixed_field> Severity	Unsigned byte	One	False	Value set, offset=false, ranges/enums: 0= NONE 1= INFO

## Unclassified

					2= WARN 3= ERR 4= FAIL [5..255] = <reserved>
3	<fixed_field> Code	Unsigned byte	One	False	Value set, offset=false, ranges/enums: 1= PROCESSOR 2= RAM 3= ROM 4= FILESYS 5= POWER 6= SENSOR 7= ACTUATOR 8= SOFT [9,254] = RESERVED 255= BUSY
4	<bit_field> Logtime	Unsigned integer	One	False	Bits 0..9, Value set, offset=false, ranges/enums:[0,999] Bits 10..15, Value set, offset=false, ranges/enums:[0,59] Bits 16..21, Value set, offset=false, ranges/enums:[0,59] Bits 22..26, Value set, offset=false,
5	<variable_length_string> Descriptor	Variable length string (byte[])	One	False	(Length min..max = 0..32)

### 3.6.5 Protocol Behavior



**Figure 3.6-2: HEALTH REPORTER SERVICE PROTOCOL BEHAVIOR**

**Table 3.6-5: HEALTH REPORTER SERVICE STATE TRANSITIONS**

Label	Trigger	Conditions	Actions
<b>A</b>	QueryHealthStatus		sendReportHealthStatus
<b>B</b>	PerformCBIT	isControllingClient	performCBIT

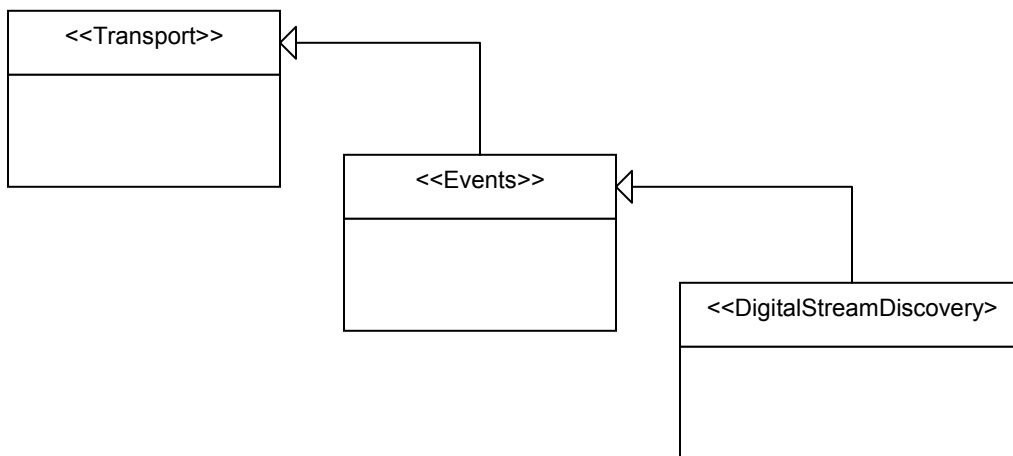
**Table 3.6-6: HEALTH REPORTER SERVICE TRANSITION ACTIONS**

Action	Interpretation
sendReportHealthStatus	Sends a ReportHealthStatus message to the requesting client.
performCBIT	Initiates a Built in Test on the node/component receiving the command that updates the health status that is reported by subsequent ReportHealthStatus messages.

### 3.7 Digital Stream Discovery

name= DigitalStreamDiscovery  
version=0.1  
id=urn:jaus:iop:DigitalStreamDiscovery

Inherits-from Events  
name=events  
id= urn:jaus:jss:core:Events  
version=1.0



**Figure 3.7-1: DIGITAL STREAM DISCOVERY SERVICE**

### 3.7.1 Description

The Digital Stream Discovery Service provides a mechanism for SAE JAUS-based components to discovery network entities that transmit digital data streams (usually video and/or audio) in a standards-compliant format. Because of the wide-spread support for numerous streaming standards, this service does **not** propose a JAUS-specific format for data; it only provides a discovery mechanism based on Uniform Resource Locator (URL).

### 3.7.2 Assumptions

Messages may be delayed, lost or reordered.

### 3.7.3 Vocabulary

**Table 3.7-1: DIGITAL STREAM DISCOVERY SERVICE VOCABULARY**

Message ID (hex)	Name	Command
<b>Input Set</b>		
E8A2	<a href="#">QueryDigitalStreamEndpoint</a>	False
E8A3	<a href="#">RegisterDigitalStreamEndpoint</a>	False
<b>Output Set</b>		
F8A2	<a href="#">ReportDigitalStreamEndpoint</a>	false

### 3.7.4 Encoding

#### 3.7.4.1 Input Set

##### 3.7.4.1.1 ID E8A2: *QueryDigitalStreamEndpoint*

Queries for a list of known stream source endpoints.

**Table 3.7-2: QUERY DIGITAL STREAM ENDPOINT MESSAGE ENCODING**

body └─ <empty>
--------------------

##### 3.7.4.1.2 ID E8A3: *RegisterDigitalStreamEndpoint*

Registers a stream with the service, or removes an existing stream. Each endpoint is represented by a URL; however, the URL shall not require a Domain Name Service (DNS) to resolve. In addition, each stream may also specify a JAUS ID that hosts additional SAE JAUS Services for the configuration and control of the stream, as well as a SensorID that identifies the stream source.

## Unclassified

<div style="margin-left: 100px;"> body <div style="margin-left: 20px;"> └─ <b>list</b> name = RegisterDigitalStreamEndpointList  (count_field = unsigned short integer)  <b>sequence</b> name = RegisterDigitalStreamEndpointSeq  <b>record</b> name = RegistrationTypeRec  <b>record</b> name = EndpointCommonDataRec </div> </div>					
<b>record</b> name = RegistrationTypeRec					
Field	Name	Type	Units	Optional?	Interpretation
1	<fixed_field> RegistrationType	Unsigned byte	one	False	Value_enum: 0 = Register 1 = Unregister
<b>record</b> name = EndpointCommonDataRec					
Field	Name	Type	Units	Optional?	Interpretation
1	<presence_vector>	unsigned byte	one	false	
2	<fixed_field> StreamType	Unsigned byte	one	false	Stream type  value_enum: 0 = RTSP 1 = MPEG2-TS 2 ... 255 Reserved
3	<variable_length_string> StreamURL	Count_field = unsigned byte	One	False	URL of the source of the stream. This URL should not require a DNS to resolve; hence, an IP address should be substituted for a host name.
4	<bit_field> JAUS_ID	Unsigned integer	one	true	JAUS ID of the component that hosts any configuration and control services for this stream.  Bits 0-7: Component ID Bits 8-15: Node ID Bits 16-31: Subsystem ID
5	<fixed_field> SensorID	unsigned short integer	one	true	The ID used by the configuration and control service to identify this stream source.

### 3.7.4.2 Output Set

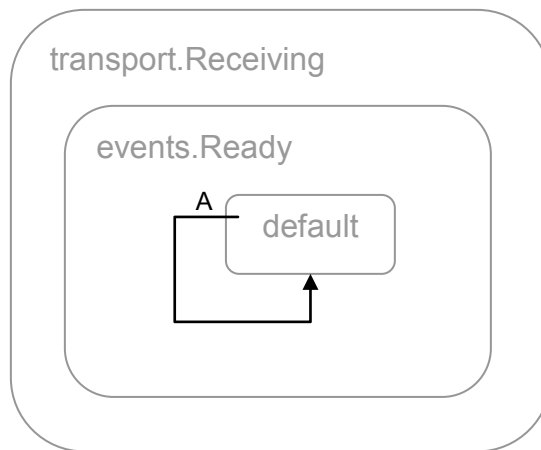
#### 3.7.4.2.1 ID F8A2: ReportDigitalStreamEndpoint

Reports a list of known stream sources. Each endpoint is represented by a URL; however, the URL shall not require a Domain Name Service (DNS) to resolve. In addition, each stream may also specify a JAUS ID that hosts additional SAE JAUS Services for the configuration and control of the stream, as well as a SensorID that identifies the stream source.

**Table 3.7-3: REPORT DIGITAL STREAM ENDPOINT MESSAGE ENCODING**

body └─ <b>list</b> name = ReportDigitalStreamEndpointList (count_field = unsigned short integer) <b>record</b> name = EndpointCommonDataRec					
<b>Record Name</b> = EndpointCommonDataRec					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> EndpointCommonData	unsigned short integer	one	false	

### 3.7.5 Protocol Behavior

**Figure 3.7-2: DIGITAL STREAM DISCOVERY SERVICE PROTOCOL BEHAVIOR****Table 3.7-4: DIGITAL STREAM DISCOVERY SERVICE STATE TRANSITIONS**

Label	Trigger	Conditions	Actions
A	QueryDigitalStreamEndpoint		SendResponse message 'ReportDigitalStreamEndpoint'
	RegisterDigitalStreamEndpoint	isRegisterType	RegisterEndpoint
	RegisterDigitalStreamEndpoint	! isRegisterType && endpointExists	RemoveEndpoint

**Table 3.7-5: DIGITAL STREAM DISCOVERY SERVICE CONDITIONS**

Action	Interpretation
--------	----------------

## Unclassified

isRegisterType	True if the message that triggered this transition specifies the RegistrationType as 'Register'
endpointExists	True if the message that triggered this transition specifies an endpoint that exists in the list of known endpoints

**Table 3.7-6: DIGITAL STREAM DISCOVERY SERVICE TRANSITION ACTIONS**

Action	Interpretation
SendResponse	Send the specified response message
RegisterEndpoint	Adds the specified endpoint to the list of known endpoints
RemoveEndpoint	Removes the specified endpoint from the list of known endpoints

### 3.8 Preset Pose Service

name= Preset

version=0.1

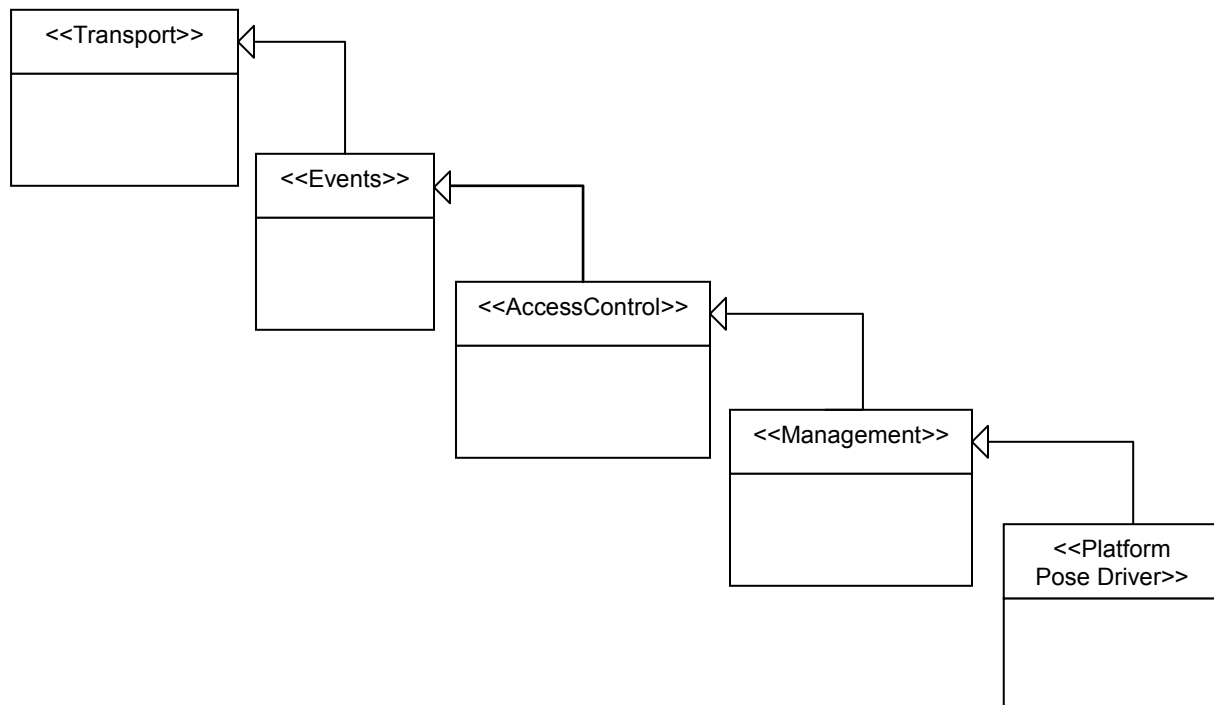
id=urn:iop:platform:PresetPose

Inherits-from Managment

name=events

id= urn:jaus:jss:core:Management

version=1.0



**Figure 3.8-1: PRESET POSE SERVICE**



### 3.8.1 Description

The Preset Pose service supports querying and setting of preset poses. For example, a Stow pose would be a configuration of platform manipulators and other actuators that best prepare it to be stowed away.

### 3.8.2 Assumptions

Messages may be delayed, lost or reordered.

### 3.8.3 Vocabulary

**Table 3.8-1: PRESET POSE SERVICE VOCABULARY**

Message ID (hex)	Name	Command
<b>Input Set</b>		
FFFC	QueryPresetPoses	False
FFFD	SetPresetPose	False
<b>Output Set</b>		
FFFE	ReportPresetPoses	false

### 3.8.4 Encoding

#### 3.8.4.1 Input Set

##### 3.8.4.1.1 ID FFFDh: *SetPresetPose*

This message sets the desired preset pose. It will cause all the components with preset poses to move to those positions.

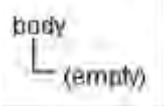
**Table 3.8-2: SET PRESET POSE MESSAGE ENCODING**

<div style="text-align: center;"> <b>body</b>            L record name = PoseRec         </div>					
<b>Record Name = PoseRec</b>					
Field #	Name	Type	Units	Optional	Interpretation
1	<fixed_field> Pose	unsigned byte	One	False	Enum 0: "Stow" Enum 1: "Deploy" Enum 2: "Drive"

##### 3.8.4.1.2 ID FFFCh: *QueryPresetPoses*

This message provides a way for a client to get information about the preset poses that the platform supports. The response to this query is ReportPresetPoses messages.

Table 3.8-3: QUERY PRESET POSES MESSAGE ENCODING


Empty message body

### 3.8.4.2 Output Set

#### 3.8.4.2.1 ID FFEh: ReportPresetPoses

This message provides the preset poses that are supported by the platform.

Table 3.8-4: REPORT PRESET POSES MESSAGE ENCODING

Body └ SupportedPresetPoses					
<b>Record</b> name = SupportedPresetPoses					
Field #	Name	Type	Units	Optional	Interpretation
1	<bit_field> SupportedPoses	unsigned integer	one	false	Bitfield of supported poses. If the bit is high ("1"), the pose is valid for this platform. Bit 0: "Stow" Bit 1: "Deploy" Bit 2: "Drive"

### 3.8.5 Protocol Behavior

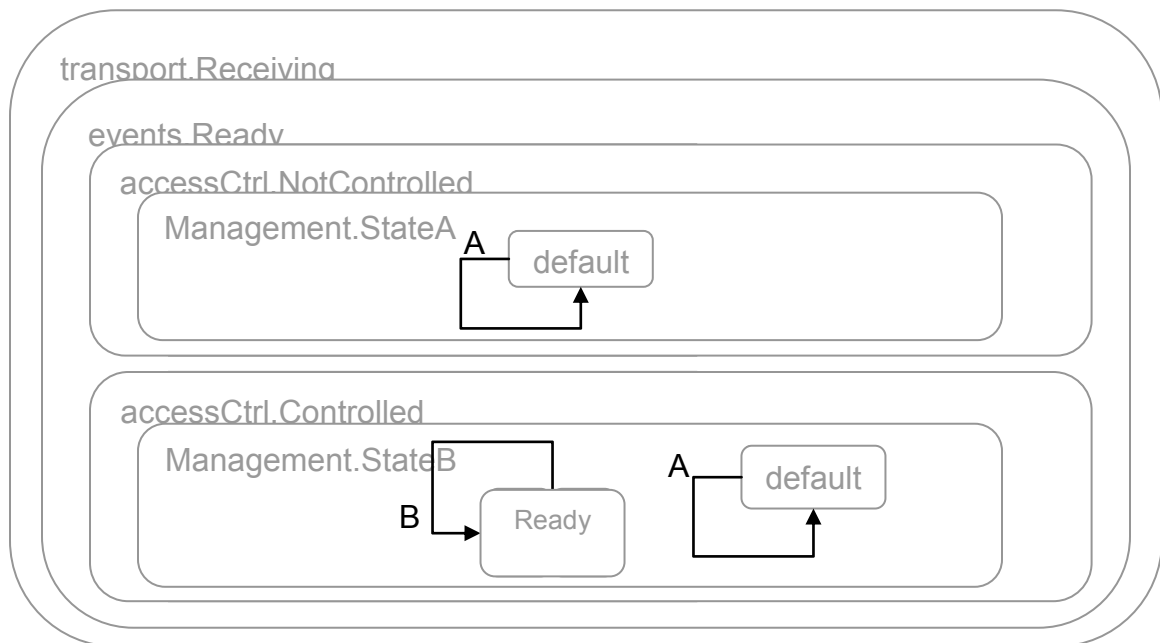


Figure 3.8-2: PRESET POSE SERVICE PROTOCOL BEHAVIOR

**Table 3.8-5: PRESET POSE SERVICE STATE TRANSITIONS**

Label	Trigger	Conditions	Actions
<b>A</b>	QueryPresetPoses		sendReportPresetPoses
<b>B</b>	SetPresetPose	isControllingClient && isSupportedPose	setPresetPose

**Table 3.8-6:PRESET POSE SERVICE CONDITIONS**

Action	Interpretation
isControllingClient	True if the requester of action is the controlling client.
isSupportedPose	True if the command message that triggered the transition commands a pose supported by this platform

**Table 3.8-7: PRESET POSE SERVICE TRANSITION ACTIONS**

Action	Interpretation
sendReportPresetPoses	Sends a ReportPresetPoses message with a list of all supported preset poses.
setPresetPose	Utilizes platform actuators and manipulator to configure system into specified pose.

## 4 Custom Messages

There are currently no custom messages defined.

## 5 Custom Transports

For the UGV IOP, transports are specified in accordance with SAE JAUS AS5669A, the JAUS/SDP Transport Specification. Transports that are specified outside of AS5669A or AS5669A transport modifications, approved for use within the UGV IOP will be specified in this section. As of this version of the IOP, there are no defined custom transports approved for use within the UGV IOP.

## **6 Conformance and Validation Requirements**

Specification and implementation of any message and/or transport defined within this document shall be done so in accordance with the exact specification as described within Section 4 (custom messages) and Section 5 (custom transports) as applicable. Implementation of messages and/or transports defined within this document shall be tested via demonstration methods to ensure that required functions and capabilities have been implemented in accordance with the corresponding specification.

## **7 Appendix A – Acronyms and Abbreviations**

ID	Identifier
IOP	Interoperability Profile
JAUS	Joint Architecture for Unmanned Systems
SAE	Society of Automotive Engineers
UGV	Unmanned Ground Vehicle